Newsletter

April 1992

In this issue...

- *Inside the Amiga 600*
- *Workbench 2.0 Unveiled*
- *Fidonet Points Revealed*
- *Machines That Can See*
- *Compression and Encryption*

HONORARY CUGI OFFICIALS

Chairman	Geoffrey J. Reeves
Treasurer	Brian Ward
Secretary	Leon Hurst
Membership Secretary	Tom Kinsella
Amiga Librarian	Rocco Matassa
Newsletter Editor	Eddy Carroll
Merchandising Officer	Aidan Hollinshead
Discounts Officer	Geoffrey J. Reeves
Librarian	Declan McArdle
PC Librarian	Brian Ward
C64/128 Librarian	Geoffrey J. Reeves

CUGI meeetings are held every second Friday, at 8:00 PM in the computer room at St. Andrew's College, Booterstown. All correspondence should be addressed to:

CUGI
c/o St. Andrew's College,
Booterstown,
Blackrock,
Co. Dublin.

All articles in this newsletter are copyrighted by the author unless otherwise stated. If you wish to reprint an article, write to its author c/o CUGI at the above address.

Newsletter

Contents

Volume 4 Number 2
April 1992

Editorial	Eddy Carroll	2
Compression Techniques	Stephen McGerty	3
Seeing Machines	Kenneth M. Dawson	5
Fidonet Points Revealed	James Ruane	7
Commodore's New Baby	Eddy Carroll	11
Hidden Secrets	Stephen McGerty	15
Workbench Gets a Facelift	Geoffrey J. Reeves	18
Diary Dates	Secretary	21

Editorial

Well, spring is here and with it, yet another issue of this journal arrives. Thanks to Geoff for taking over the January issue so ably, and at such short notice. I seem to be making a habit of going to America at just about the time another newsletter is due; purely co-incidence, of course. . .In any case, it was a worthwhile trip because I got to spend some hands-on time with the famed Video Toaster. It was *very* impressive! It's a real shame they haven't got around to doing a European PAL version yet—it would sell like hot cakes over here.

Since January, Commodore have released a new Amiga, namely the A600. A lot of people were hoping to see some higher-end machines announced at Hannover, but it seems it was not to be. Hopefully, Fall Comdex will see the launch of some (genuinely) new products to the Amiga lineup. In the meantime, the A600 is probably going to sell like hot cakes. See elsewhere in this issue for a more detailed look at it.

Interestingly, Commodore is now pushing the CDTV as a full computer system, a complete reversal of their attitude a year ago, when they warned dealers not to use the word Amiga in CDTV adverts. You can now get a sexy black keyboard, cordless mouse, and external drive, to match your CDTV. And of course, you can now add CDTV capabilities to your existing Amiga via the new A570 CD-ROM drive. As you may know, the club recently purchased a unit, and we now have the Fred Fish collection on CD-ROM up to disk 600; very useful indeed.

Apologies for the somewhat slim size of the newsletter this month. All I can say is that if you don't submit articles, I can't print them. Without your support, the newsletter will continue to shrink, so please take the hint and get writing in time for the July issue!

Now a request of a different kind. One of our members, Robert Chan, is looking for someone to help him with a research project he's working on at St. James Hospital. The project involves monitoring patients blood pressure using some specialised equipment which can be hooked up to an Amiga via an RS-232 link. The equipment sends out ascii data a few times a minute giving the current pulse rate and high/low max blood pressure reading. What Robert needs is a way to save this incoming data to disk on his Amiga 500, and then display it graphically on the screen—nothing too fancy. If your programming abilities are reasonable and you have a bit of time to spare, give Robert a ring on 537941 (work) or 935003 (home).

That's it for now. Enjoy this issue, and get writing some articles!

E.C.

Compression—The Minimalist's Goal

by Stephen McGerty

When it comes to storing or moving data, we all like to be minimalists. Whether you are squeezing a bit more onto your 105 MByte hard drive, or uploading a 200K file at 2400 baud, you want to make your data as small as possible. The advantages are obvious: small chunks of data take up less room, and are easier to move around.

Data is just a sequence of numbers, or bytes. Each byte holds a number between 0 and 255. A typical chunk of data looks like this:

```
1, 255, 6, 15
```

Nothing spectacular really. The thing to realise is that absolutely any information can be reduced to a sequence of meaningless numbers: graphics, sound, computer programs, music, text, poetry, etc.

Now, there are a number of ways of compressing data. The most obvious is to locate any repetition, and express it in a shorter form. Even simple repetition of a single byte can be useful. For example, if your data is:

```
3, 5, 4, 4, 4, 4, 4, 4, 3, 2
```

What do you notice? There are six 4's. So why not rewrite this as:

```
3, 5, (6), 4, 3, 2
```

The brackets indicate that the number is a *count* and not a real number. But how do you put brackets around a byte? How do you 'mark' a byte?

There are a number of ways of doing this, but one method I have used is to always start every sequence of bytes with a count and an indicator. The indicator says whether there is a repetition of bytes following, or a sequence of random bytes. The count indicates how many repetitions there are, or how long the random sequence is. Our example would become:

```
(2 random), 3, 5, (6 repetition), 4, (2 random), 3, 2.
```

The decompresser would start at the begining, and recreate the two random bytes, which are given after the indicator. Once 3 and 5 have been read, it takes the next byte as another indicator byte. This time it says that there is a 6 byte repetition. The number to repeat is after the indicator (4). Finally, another indicator is read, and the last two random bytes are extracted.

The indicator byte is simply a normal byte. The most significant bit is used as the random/repetition flag, and the other 7 bits hold the actual count. The problem with this, is that the count must be less than 128. So if there are 130 repeated 4's, then the sequence will have to be:

(127 repetitions), 4, (3 repetitions), 4.

Now this method of compression is very simple. There are lot of cases where it will produce 'compressed' data bigger than the original data. However, for certain types of graphic data (those with lots of blank areas) it compresses quite well. The main advantages of this method are simplicity, and speed. The algorithm is not too complex, and most implementations are very fast. In fact, the familiar `IFF` pictures you see on the Amiga are stored on disk using this compression method, which is known as run-length encoding.

The other compression techniques are a good deal more complex. The commonly used Huffman method is very good at compressing English text, as it works well with data which has uneven frequencies of characters (i.e. 'e' is more common than 'z'). The original *arc* program used this method to compress files.

The Lempel-Zev technique is also much more complex than run-length encoding. It builds up a list of strings, and expresses the data in terms of these strings. A simple example is the sequence 'the' which appears very frequently in English text. If that was stored as a single entry, there would be significant savings every time 'the' was encountered.

Most of today's compression utilities like *lharc* use a combination of Lempel-Zev and Huffman encoding to obtain their high compression rates, which is where the `.LZH` file suffix arises from. Next time, we'll look at Huffman encoding in more detail.

For Sale

One 512K Amstrad PCW 9512 word processing system with letter quality printer, only lightly used. Price in the region of £350.00. Ring Eddy Carroll on 287 4540 if you're interested.

Seeing Machines

by Kenneth M. Dawson

Dept. of Computer Science, Trinity College Dublin

If the picture that science fiction gives, of robots wandering around in the world, is ever to be realised, we must solve many complex, technical problems. These problems fall under the broad heading of the development of anthropomorphic systems (i.e. systems that simulate some aspect of human behaviour), and may be classified into three main areas: perception, cognition and action.

Perception, Cognition and Action

'Action' for a machine is studied through the field of robotics, and more complex robot manipulators are appearing almost every day. Robots can now perform tasks ranging from high precision industrial assembly to such complex tasks as playing the piano!

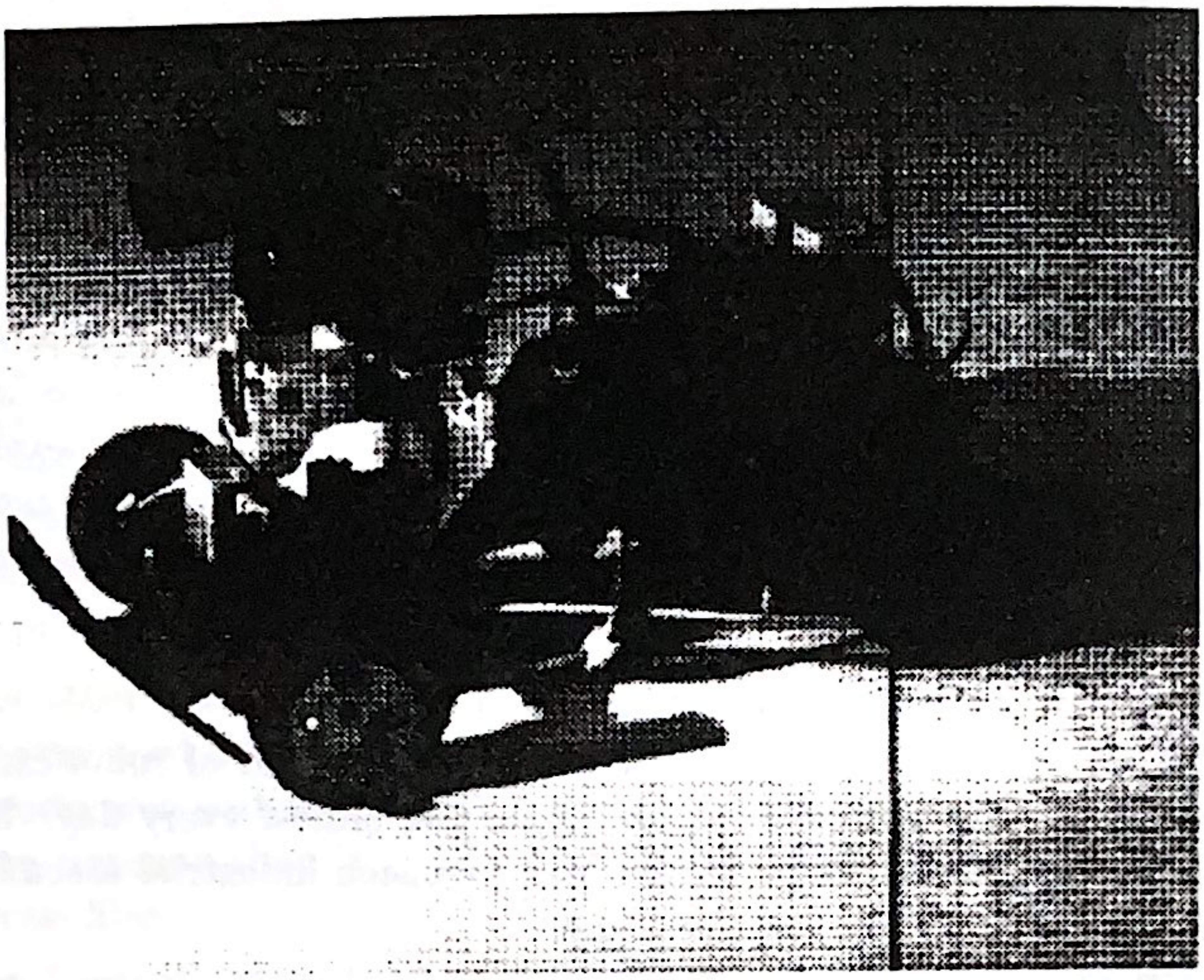
'Cognition', or understanding/reasoning, is addressed through the study of artificial intelligence and cognitive science (the latest buzz-term). Additionally, over the last ten years, the subject of building networks which employ mechanisms similar to those of the human brain (i.e. neural networks) has been addressed. However, as the human brain is extremely complex and not well understood, these attempts have only demonstrated a shadow of the potential/power that the human brain exhibits.

'Perception' is the mechanism for obtaining information from the world around us. We employ many senses: sight, hearing, smell, touch and taste; and of these senses, sight, hearing and touch are the subjects of much ongoing research in the area of automation. Of these three senses though, sight/vision is undoubtedly the richest, providing a vast amount of information about the environment which surrounds us.

Machine Vision

Vision is an extremely complex task, although to us, as humans, it appears quite straight-forward. Humans have the most powerful existing vision systems, but it was not until scientists attempted to provide these capabilities to machines that we began to understand how difficult the task actually is.

The automation of vision has been the subject of much research around the world since the 1960's. However, presently we are capable of developing systems only to address very constrained versions of the problem. We can reliably inspect bare printed circuit boards in order to ensure that the copper tracks have been laid correctly, but reliably inspecting a populated



A wandering robot designed in Trinity College

board (i.e. one that has chips mounted on it) is at the very edge of our current avilities. The reasoning behind this difference is the number of spatial dimensions which must be considered.

The bare printed circuit board may be considered as a two-dimensional object, and as images are two-dimensional as well, the inspection is reasonably straightforward. However, a populated printed circuit board is three-dimensional, and yet the images are still two-dimensional. Somehow, we have to recover from the massive loss of information.

Robot Vision

Performing industrial inspection is the main use of machine vision at the moment (and is what pays the bills!). However, in the long term, we are more concerned with the development of general-purpose vision systems—systems which can understand any three-dimensional scene (from an office to a field) and hence allow robots to move around the world in a well-behaved manner.

Since research in this area began, we have advanced quite significantly and many sub-problems have been successfully solved (such as the use of two cameras to work out how far away things are, in the same way that our eyes perform that task). Robot systems which are capable of wandering around and not hitting anything (too often) have been developed (e.g. see picture). Most of these systems work work only in constrained environments, such as an office or factory, but the significant fact is that we are moving rapidly towards the commercial exploitation of complex remote robots (i.e. robots which are not physically attached to anything).

The Future

So what of the future? What will the machines be doing fifty years from now? Well, looking at the fields of robotics and vision as they currently stand, we can be reasonably sure that we will see robots wandering around performing useful tasks—collecting rubbish, delivering mail, cutting the grass, and who knows what other activities? All of these things are almost within our grasp, and as technology becomes even less expensive, the future (in terms of technology) will, most probobably, arrive at a phenomenal rate.

Fidonet Points Revealed

by James Ruane

Fidonet started in the early 1984 when a BBS sysop in the US thought that it would be nice if users on his BBS could exchange messages with users on other systems. And so it was that he wrote the first Fidonet mailer and BBS, called *Fido.* At night, Fido would pack the mail and send it to another system. Fido on the receiving side would unpack it and pass it to the BBS users.

The system took off like wildfire. In three months, there were 50 systems participating, and by 1985, the network, now called *Fidonet*, had 150 participating BBS's, known as *nodes.* There are now more then 10,000 nodes in every corner of the world.

Like Usenet, Fido has its own method of addressing mail. A four-part address system was developed:

`Zone:Node/Net.Point`

So my address is `2:263/198.2` which can be read as:

`Europe:Ireland/DecBBS.Jamie's Point.`

(Note that the .point part is optional.)

What Does Fido Offer?

NetMail. Private EMail between users of different BBS, anywhere in the world. You usually have to pay the Sysop to get *NetMail Credits* and each message is then charged from your supply of credits. Credits usually cost a penny each, and a Netmail message to the UK might cost you 50p. The BBS routes your mail to its destination through other boards.

Echomail. Echomail is public, international mail. It is rather similar to the conferences or SIG's on public BBS, except that each BBS shares messages with its neighbours every night. Postings to Echomail areas are usually free. There are some famous personalities on Fido; for example, Stewart C. Russell (of Amiga Shopper) inhabits Amiga UK.

There are a huge number of areas, ranging from Cookery to Amiga to Startrek. There are two Irish echoes: *Comhra* (An Echo as gaelige) and *Craic* (General chatter). There was a third, *Dublin 91*, but I believe it is no longer operating as the Year of Culture has ended.

Files. Fido also has a facility to distribute files. There are a number of file distribution nets carrying Amiga software only. You send your program to the coordinator and within a few days your file will be available all over the world.

How Do I Gain Access To Fido?

Just log onto any BBS that supports Fido, such as:

TOPPSI	2:263/151	711 047
Nemesis	2:263/150	324 755
DecBBS	2:263/198	288 2454
RA Dublin	2:263/172	322 249
ICON	2:263/152	971 660
AmigaGate	2:263/153	820 5135

Alternatively, if you plan on getting heavily into Fido, you could become a *point.* This involves dialling into the host BBS you have registered as a point with, and downloading the messages. As the Amiga areas generate about 500K of messages each week, it helps to have a high speed modem, to keep your phone bill (and Telecom's profits) to a minimum. The software you need is as follows.

Trapdoor

Trapdoor is a Fidonet mailer and as such a decendant of the original "Fido" mailer. It is (in my opinion) the best Fidonet mailer for the Amiga and is feature-filled. These features mean that it is (unfortunately) shareware, with a fee of 350 Austrian Schillings (about £20). It will act as a *frontdoor* for a BBS or a *tosser* for a point. In BBS mode it can launch the BBS if a human caller is detected, using AmigaDOS or ARexx, or else transfer mail directly to a point. (You can tell when a BBS has a mailer running, since it will display a message such as "Press ESC to enter the BBS" when you first connect). Trapdoor has full support for the Zone Mail Hour; this is a special hour in the middle of the night (02:30–03:30 AM) when logons are disabled, so that all the Irish Fido boards can transfer mail and messages between themselves.

In point mode, you simply enter the phone number of your *Boss* (Host BBS) into the configuration file, then start Trapdoor, and it does the rest. For example, you could set it up using ARexx to dial your boss on Wednesdays and Saturdays at midnight, download and upload mail, and maybe even download files.

Another function of Trapdoor is Fidonet *File Requests* (FReqs). These are special files which either you or Foozle (create, named with the BBS's Fido number, and containing the files you want, like this:

```
File 2:263/198.0.REQ containing:
     Csh519.Lha
     PopUpMenu.Lzh
     Files
```

When Trapdoor reaches 2:263/198.0 (DecBBS) it will request and download the files contained within the .REQ file. When you next check your Mail:Inbound directory, the files should have appeared. Oh, and technical support for this program is available through the *Trapdoor* echo. It's moderated by the authors, which means you get a quick response to any questions you might have.

However, all this mail is useless without a mail manipulator to let you read and reply to it. Which brings us onto our next topic.

Foozle

Foozle, by Peer Hasselmyer, is 'a fast Fidonet-compatible message-editor /tosser/scanner/packer'. What that means is that it is an interface to all the tools required to read, write, and access Fidonet Mail packets. When you first get Foozle, you run the *FZI* program. This is an installer/configuration editor. With it, you set the areas you receive, what BBS you receive them from, the text editor you use, the file archiver, etc.

With all that done, just invoke Foozle. When Foozle starts, there will be no messages in the message areas, as you haven't selected *Import* yet. When you select "Import", Foozle searches the MAIL:Inbound directory for any mail files. If they are compressed, Foozle uses the unpacker you specified to unpack them (make sure you're using the right one!) then splits it up into individual messages which are then put into the right areas.

On floppies, this procedure can take quite some time, not to mention that you'll be rather short on storage space for messages (remember that half meg of messages per week from the *Amiga* area...) and may even run out. From RAD: however, the import rarely takes more then a few seconds. Incidentally, the author recommends that the system be used with at least 1 Mb of RAM and a hard disk.

After unpacking the messages, you can begin reading mail and writing replies. You use the cursor keys to move around the message base. If you want to write replies or post messages, you need an external editor. Foozle starts it up with the text of the message you're replying to already loaded, and you write your reply/message. Select *Save*, click a requester in Foozle, and hey presto, your message has appeared. It's as simple as that.

You can also send Netmail through Foozle, as well as manage file requests (much simpler then all that messing with a text editor). Foozle also supports a *Userlist*. If you set up a file with the names of people and their Fido addresses, when you enter their names in the mail requester, the address is automatically entered.

When you're done, simply select *Export* and Foozle will archive your mail and pop it in the MAIL:Outbound directory ready for Trapdoor to send it to your boss.

There are many other features available in Foozle: *Area Rescan* if your message base gets damaged, *Maintenance* to delete a certain number of messages, *Report* to report the status of your message base, and *Print* to print a message. There is also an ARexx port, to help you automate common tasks.

Where can you get Fido Software?

Foozle is available from DecBBS as FOOZLE10.LZH, and the update to V1.1 is FZ101UPD.LZH. Trapdoor is also available as TD_1_80.LZH.

All in all, the Foozle/Trapdoor combination isn't that hard to setup. In fact, it's quite easy to do so. The manuals are clear and there are Foozle and Trapdoor both include sample configuration files. The entire Fido network and point setup is quite easy to use; you don't need to be a comms guru or anything.

That just about wraps up my overview of Fido. If you have any questions, contact me at 2:263/198.2 or through the club.

Commodore's New Baby: The A600

by Eddy Carroll

By now, most people will have heard that Commodore launched a new member of the Amiga family at the CeBit show in Hannover last March, said member being the Amiga 600. Since some reports of its capabilities have been somewhat confusing, I thought I'd have a go at clearing things up somewhat (and hopefully avoid adding to the confusion).

Something Old, Something New

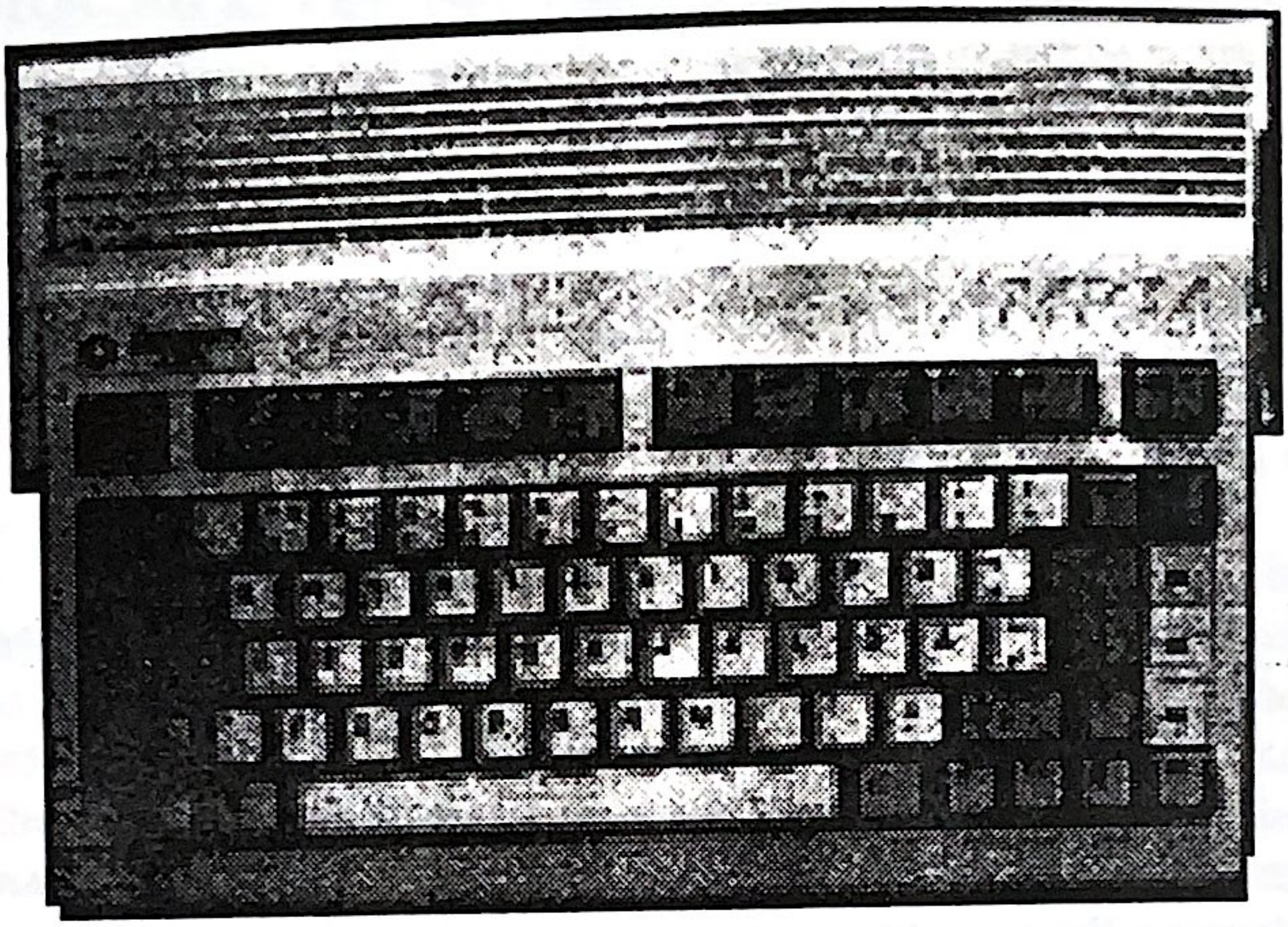
The first thing you notice about the A600 is the case, which looks for all the world like the mutant half-brother of the Commodore 64. To achieve this small footprint, some reasonably drastic changes were made to the A500 design. Most obviously, the numeric keypad has been dropped, and the cursor keys have been moved into the main keyboard area, positioned just below the Return key.

The 880K floppy drive has moved a little closer to the front of the case, and is now positioned at a slight angle, giving it a strange appearance. Conveniently, the mouse and joystick ports are now to be found on the right hand side of the case once more, which will save a lot of rooting around the back when you want to plug things in.

Speaking of the back, all the existing connectors are present: RGB Video, External Drive, Stereo Audio, Serial, and Parallel. The composite video output is now full colour instead of black and white, and it has been joined by an RF output to allow the A600 to be plugged directly into the back of a TV set, without having to go through an external modulator. As with the A500, the power pack remains external.

The biggest surprise (unless you were at the September Developers Conference in Milan) is the presence of a small new slot on the left side of the case, replacing the old expansion connector. This is a PCMCIA-compliant expansion interface which can accept small credit card sized peripherals. These include items such as RAM, flash ROM, modems, SCSI interfaces and so on.

The PCMCIA interface is starting to become popular on laptops and pen-based computers, so in theory, you could use it to easily transfer files from such a system to your Amiga. It is also likely that we will see games being distributed on cards at some stage in the future. However, this is unlikely to happen quickly, for two reasons: ROM's are currently very expensive to manufacture in credit card format (I've heard prices of



The new Amiga 600. What's yours called?

$150/megabyte), and in addition, the lucrative existing A500 market currently has no way to make use of such cards.

Underneath the A600 is the familiar looking trapdoor of the A500. However, you won't be able to plug in your old A501 and A501+ memory expansions, since the layout of the connector has changed a little bit. You will still be able to expand your CHIP memory using this port, and doubtless the usual range of PC emulators and mega-ram boards will surface in A600-compatible forms fairly shortly.

Under the Hood

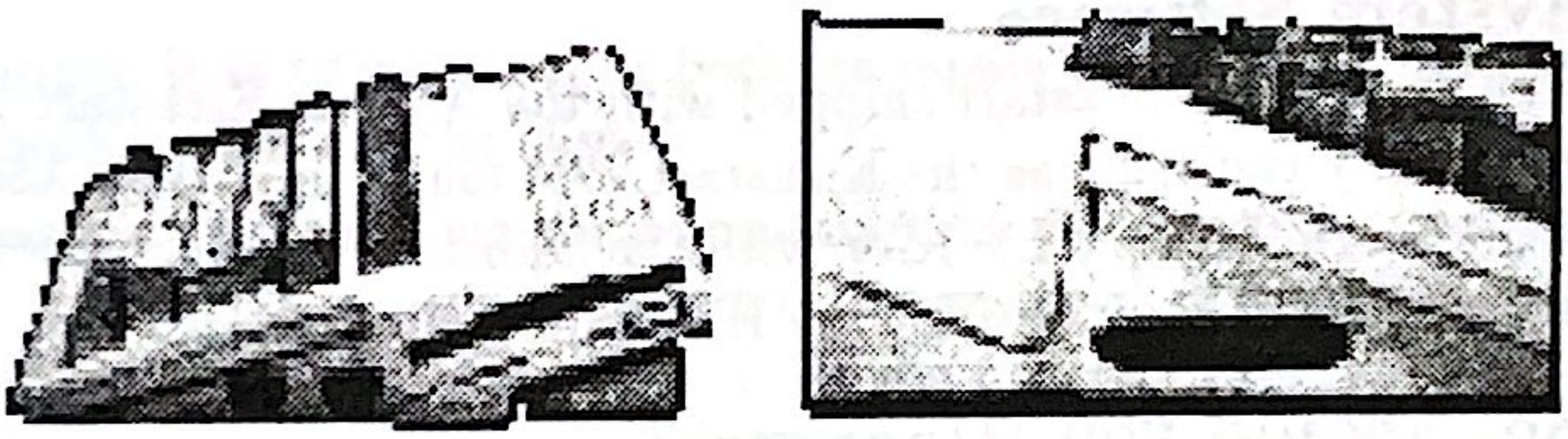
So much for the outside. How about under the cover? Well, you won't be surprised (though perhaps a little disappointed) to hear that the basic computer is almost identical to the Amiga 500. The fab three (Agnus, Paula, and Denise) are still there, though Gary has undergone a sex-change to become Gayle. One megabyte of CHIP RAM is standard, and Agnus (at this stage looking *very* bloated) can handle as much again, to bring the total possible CHIP RAM to two megabytes, as with the A500 Plus.

Unlike the A500, where most components were socketed, the A600 motherboard is entirely surface-mounted (i.e. all the chips are soldered directly onto the motherboard). This makes the computer much cheaper to manufacture, but means that there is absolutely no way to install internal upgrades, such as 68030 accelerator cards. The one exception is the Kickstart

ROM, which is still socketed, to allow for upgrades in the future.

Interestingly, even the two 8520 i/o chips are surface-mounted. If you've ever managed to blow one of these on your A500 by plugging in a cable while the Amiga was turned on, you might justifiably be wary of this, since it makes them impossible to replace at home (replacing surface-mounted components requires expensive equipment well into the four-figure range). In the A600, the 8520's are buffered, which should help make them a lot more robust; only time will tell.

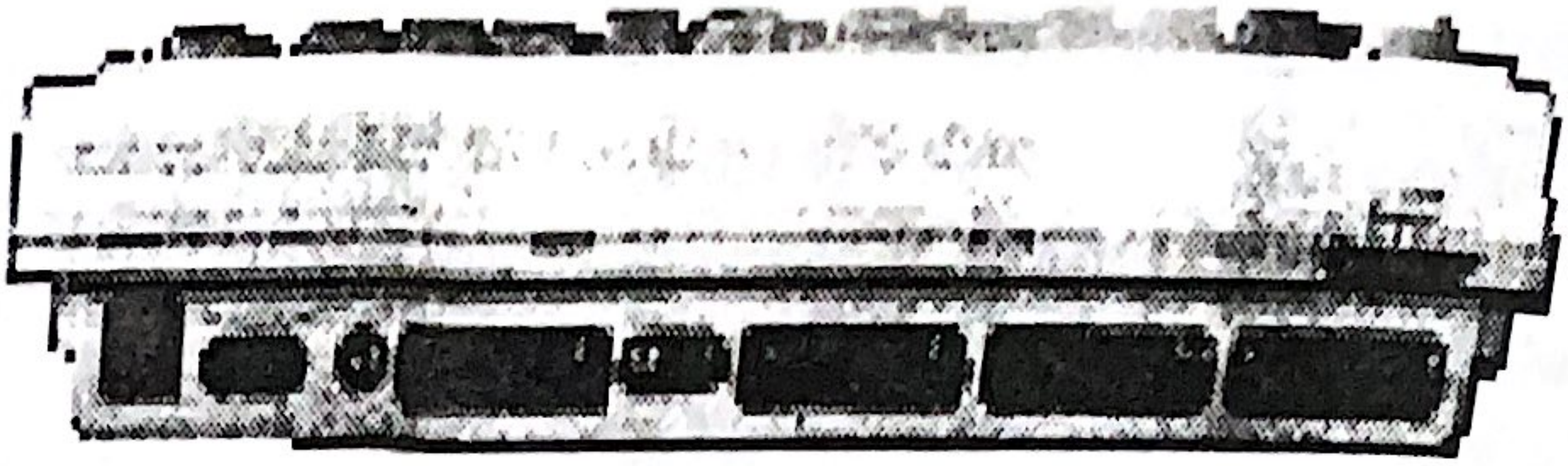
So, what's this about Gayle? Well, Gayle is a new gate array that handles all the functions of the original Gary chip, and also looks after the PCMCIA slot and IDE hard disk interface. IDE interface? Yes, indeed; Commodore have kindly provided a way to plug in an internal 2.5 IDE hard drive (which, incidentally, has its own LED on the A600 case).



The Amiga 600's two slots: floppy and PCMCIA

In fact, the A600 will be offered in two configurations: the basic A600, and the A600-HD with 20 Mb hard drive (larger sizes may also be available). If you decide to buy the basic A600 model and install your own hard drive, however, you may run into a problem. The Kickstart ROM supplied with the A600-HD contains a hard disk device driver not present in the A600 version, so you need to purchase an upgrade kit from Commodore (or have your dealer perform the upgrade for you). This was apparently done deliberately, to prevent problems arising from incompatible IDE drives. Apparently 2.5 IDE drives are not terribly standard, and not all models work reliably with the A600 interface. By controlling the upgrade path, Commodore can ensure that only the correct IDE drives are used with the A600. At least that's their story...

As for the hard drive itself, IDE may initially seem like a strange choice for an interface; after all, isn't SCSI the de-facto standard on the Amiga by now? However, given the intended market of the A600, it makes sense. The cost of the IDE controller is essentially nil, since it is integrated into Gayle, whereas a SCSI interface would have required a separate controller



The A600 sports the usual collection of ports, plus some new ones.

chip. Transfer rates from IDE drives can reach 800-900K/second, more than enough for a computer like the A600. Finally, a 2.5 drive is required, and these are much more readily available in this size than SCSI drives. Currently, capacities from 20 Mb to 120 Mb are available from various manufacturers, though the larger varieties tend to be more expensive than their 3.5 cousins. Prices will drop as demand increases.

System Software

The version of Kickstart shipped with the A600 is Kickstart 2.05. This is essentially the same as the Kickstart 2.04 found in A3000, A500 Plus, and Workbench 2.0 Upgrade Kits, with the addition of drivers for the PCMCIA slot and (for the A600-HD) the IDE hard disk interface.

So, Should you 'Upgrade'?

Most existing A500 owners are unlikely to gain any benefit from changing over to an A600. The expansion capabilities are fairly limited compared to the A500 (at the moment; this may change), with the only real benefit being the possibility of an internal hard drive.

Commodore's target seems to be the Nintendo and computerless households that are looking to get a 'real' computer for the children. Although Commodore have introduced A600 at the same pricing level as the A500, bear in mind that the A500 has dropped over £150 in price since it first became available, whereas the A600 is merely at the start of its life. Since it is much cheaper to manufacture than the A500, I would expect to see the price fall drastically over the next 18 months (and particularly in the run up to Christmas). At the moment, the A600 can be picked up in Dublin for £480 with the A600-HD going for £620.

So, now we have a new low-end Amiga. What about the high-end? Well, Commodore has two completely separate hardware groups, one dealing with low-end systems and one with high-end. We've seen what the low-end group has been doing for the past year; hopefully, the high-end group will have something to show soon as well! Watch this space...

---

w
w.
no

a c
the
char
same
in the
you ju

The
Howeve
the sam
means y

Below
file, and t
while the
of plain En

One com
letter in Eng
the key used

Hidden Secrets

by Stephen McGerty

The encryption of messages is an age old problem. In the modern world, the security of data is of vital importance. However, at the home user level, there is rarely a need for high security precautions on files. Often, obscuring obvious will be enough to deter prying eyes.

One of the simplest methods of coding a file (and then decoding it) is to use the boolean function *exclusive or* otherwise known as XOR. This function is defined as follows:

	0	1
0	0	1
1	1	0

Put simply, it is true (1) when both its inputs are different, and false (0) when both inputs are the same.

It has the neat mathematical property that x XOR y XOR $y = x$. In other words, if you XOR x with y, then XOR the result of that with y again, you will get back to the first number, x. If you understand why $-(-3) = 3$ in normal maths, then you've got the idea.

It is quite possible to use XOR as an encryption process. Just start with a codeword, then XOR the first character of the file to be encoded with the first character of the codeword. The result of the operation is the first character of the coded file. Then take the second character, and do the same thing, and so on for the whole file. When you run out of characters in the codeword, (which is going to be quite short compared with the file) you just start back at the first character of the codeword again.

The result of this is a very different file to the one you started with. However, because of the special property of XOR, you simply apply exactly the same process to the encoded file, to get back the original file! This means you only need one routine to do both the encoding and decoding.

Below is a short program, called *StvCrypt*, which allows you to encode a file, and then decode it later on using the same keyword. Be careful though; while the encrypted file will deceive most casual browsers, XOR encryption of plain English text can be readily broken using fairly simple techniques.

One common method is to observe that the letter 'e' is the most common letter in English text, and use this information to determine the length of the key used to encrypt the text. Once the length of the key is known (or guessed), the frequencies of other letters in the alphabet can be used to help determine what values other letters have, and hence what the key used was. With this information, the file can be easily decoded.

One simple and effective way to get around this is to compress the file before encrypting it, using *compress* or *lharc*. This effectively removes all redunancy from the file, making it much more difficult to crack it using standard techniques. *compress* (a Unix utility, but also available on the Amiga) is probably preferable to *lharc* since *lharc* files tend to start with the unique identifier `-LH1-` or `-LH5-`. If a would-be decrypter knew the file was an encoded *lharc* file, they could easily determine 5 characters of your codeword!

```
/*
 *    StvCrypt (c) Copyright Stephen McGerty 1992
 *
 *    Usage: stvcrypt filename[.SCD] codeword
 *
 *    If FileName has an .SCD extension then the file will
 *    be decoded and a file with no .SCD extension will be
 *    produced. Otherwise, a coded version of FileName will
 *    be produced with an extension of .STC.
 */

#include <stdio.h>
#include <stdlib.h>
#include <string.h>

int main(argc, argv)
int argc;
char *argv[];
{
    FILE *fp, *nfp;
    char *FileName;
    char *CodeWord;
    char *dotpos;
    char NewFileName[30];
    int  Source, Dest;              /* Read and Write bytes */
    int  CodeWordCount, CodeLen;    /* For the XOR of dest  */

    printf("StvCrypt. By Stephen McGerty (c) 1992.\n");
    if (argc != 3) {
        printf("Usage: stvcrypt filename[.SCD] codeword\n");
        exit(0);
    }

    FileName = argv[1];
    CodeWord = argv[2];
    fp = fopen(FileName, "rb"); /* 'b' is binary for PC's */
    if (fp == NULL) {
        printf("Error opening input file %s.\n", FileName);
        exit(0);
    }
    dotpos = strrchr(FileName, '.'); /* Find dot in name */
    if (dotpos != NULL && stricmp(dotpos, ".SCD") == 0) {
        *dotpos = 0;
        strcpy(NewFileName, FileName);
    } else {
        strcpy(NewFileName, FileName);
        strcat(NewFileName, ".SCD");
    }
    nfp = fopen(NewFileName, "wb");
    if (nfp == NULL) {
        printf("Error opening output file %s.\n",NewFileName);
        fclose(fp);
        exit(0);
    }
    CodeWordCount = 0;
    CodeLen = strlen(CodeWord);
    while ((Source = getc(fp)) != EOF) {
        Dest = CodeWord[CodeWordCount] ^ Source;
        CodeWordCount = (CodeWordCount + 1) % CodeLen;
        putc(Dest, nfp);
    }
    printf("File %s created okay.\n", NewFileName);
    fclose(nfp);
    fclose(fp);
    return (0);
}
```

Workbench Gets a Face Lift

by Geoffrey J. Reeves

When I was Editor, I mentioned that I'd received my Release 2 *ROM Upgrade Pack*. While it's true that many members have a Version 2 machine, having purchased an Amiga 500+, there are some who have yet to move into the new environment. I've had access to V2 for some time now and I must say that it has always been *very* nice to work with. For the novice user, the improved control over your system is a delight. But what if you want to upgrade from V1.3 (or even V1.2)?

The Upgrade Pack

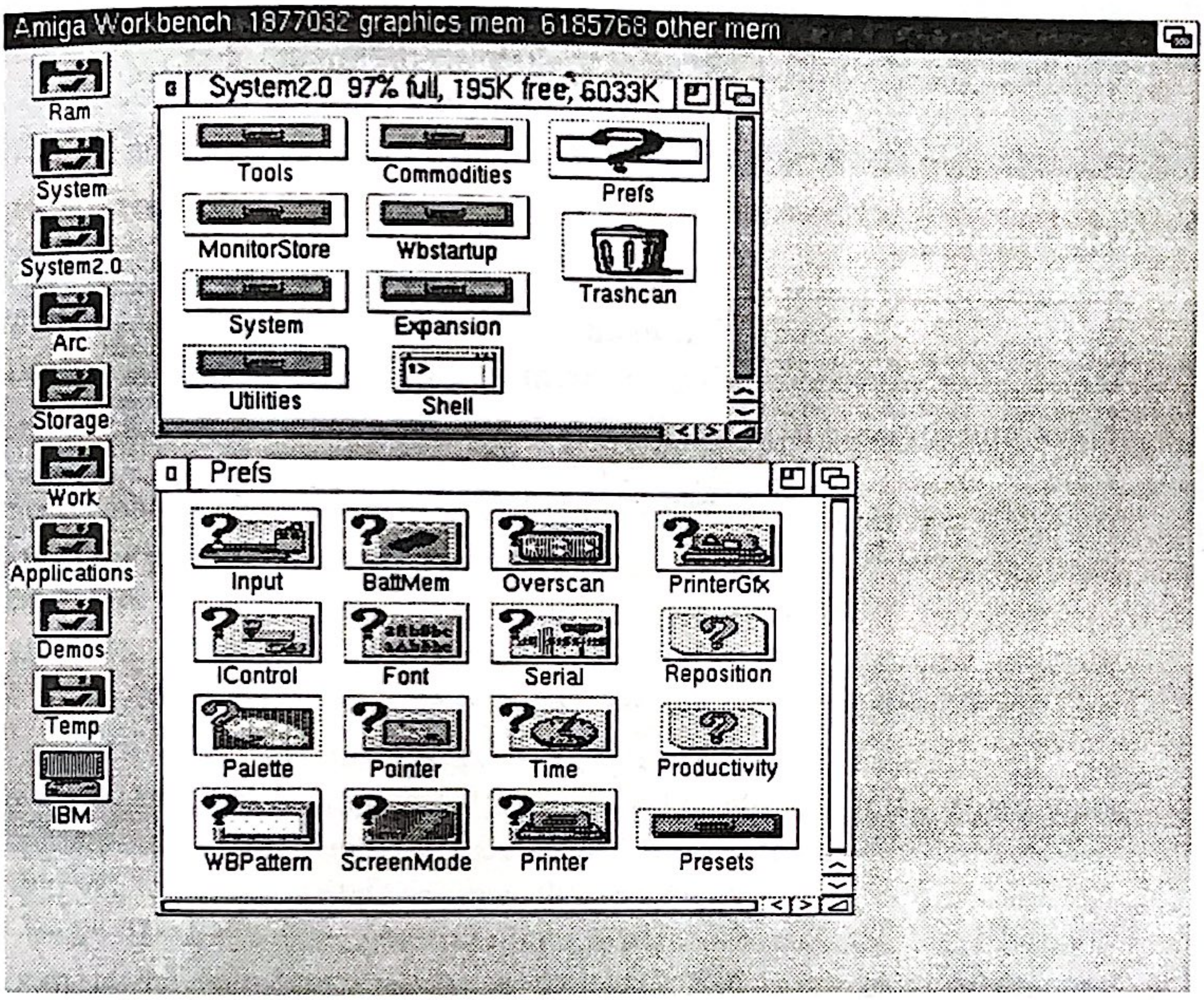
As most of you should know by now, the Upgrade Pack contains:

- KickStart ROM
- Manuals—*Using the System Software*
  Getting Started
- Four disks—*Install*
  Workbench
  Extras
  Fonts

The ROM is a 2.04 version of the KickStart ROM—there is a 2.05 being released for the Amiga 600 but that's another story. It comes with a jumper wire connecting two pins for compatibility with older motherboards. As some of you will remember from the meeting when we upgraded a computer, it helps to read the documentation—newer machines do not work if the jumper has not been removed! Putting in the ROM requires some care and opening your computer may itself be a problem; contact someone experienced if in doubt.

The *Getting Started* manual is a 48 page booklet outlining the changes from the old system, the installation of the ROM and software, and trouble-shooting hints if things don't go as suggested. It's light reading but gives you an idea of just how much thought has gone into making the new system look, feel and work better. That's not to say V2 is perfect but it is a very comfortable and easy environment in which to work.

By far the best manual from Commodore must be the *Using the System Software* tome. It's a monster-sized folder containing full details of all the aspects of V2. It even lets slip some items not included in the final release. For example, I was looking up *WBStartup* in the index and instead found *WBConfig*. Odd, I thought... didn't remember seeing that one before. So I looked up the suggested page to find a note telling me that this had been



Workbench 2.0: The look of the 90's

removed as other (nicer) features had superceded its use. Interesting; what else is hidden in this V2?

But what *is* in this monster? It is split into four main sections covering Workbench (380+ pp), AmigaDOS (300+ pp), ARexx (168 pp) and various Appendices (120+ pp)—that's about 1,000 pages in all. Compared with the smaller manual supplied with the A500+, this is *very* comprehensive. You won't need to get more reading than this for a while! The inclusion of the section on ARexx highlights its omission from the A500+ manual. Are users of that machine not expected to use ARexx? The 168 pages will give you a good overview of the language and its uses. Hopefully, we'll see more and more software supporting ARexx in the future.

The four disks include a pleasant extra—the *Install* disk. I must admit to being surprised to find that A500+/hard disk owners (and Brian) had no easy way of installing V2 on their system disk. Manually copying the various disks is not the way to have to do it—besides which one could easily end up with a mixture of V1.x and V2 files and that is definitely *not* a good thing. Those of us who purchased the upgrade pack have no such problems. (*In fact, there's a nice surprise hidden on the* Install *disk. Open the* Tools *drawer, then select* Show All Files *from the* Window *menu. Now open the* Test *drawer which just appeared and look what's inside.—Ed.)*

Release 2

I cannot claim to be an authority on V2—but I do like it. So many things so so much easier to do. For example, I recently got hold of a screen blanker which after a predetermined amount of time would cause the screen to go black and swarms of bees (okay... pixels) to randomly wander around the screen. Nice little utility, but even nicer to install—just drop its icon in the WBStartup drawer. That's all. Want to change the time or number of swarms, etc.? No problem; just click on its icon and using the *Information* option from Workbench's *Icons* menu, you are presented with a window showing amongst other things the Tool Types. You can edit these there and then and save your new settings. So simple.

I have praised *DMouse* many times for its features but with V2, its time may be up. *DMouse* offered me screen blanking (V2 supplies one), accelerated mouse movement (set in Preferences), screen cycling (use Left/Amiga-M) and auto-window selection (i.e. moving the mouse over a window activated it). The latter has been well superceded by V2. Because of the new-style windows in V2, *DMouse* becomes a nuisance as it tries to select every window you pass over. This results in a stuttery movement. V2 supplies one with a delay before selecting a window. I know that the Left-Amiga/Esc hot key is a nice feature but Workbench now gives me an *Execute command* option. Finally, window cycling isn't too important for me.

I would have added MFR (*Magic File Requester*) to the list of great new things except that its bug-free version is not the one I've got (I'm using Version 1.3b). It's a file requester similar to that of *req.library* but with the addition of resizing and just loads of features... unfortunately, it's not quite 100% yet.

But one thing I am quite confident about is that V2 has injected new life into what is an aging computer. Just a bit more speed and maybe colours would be nice but I'm happy with what I've got. I'll be even happier when I see what's on the latest Fish Disks—if the V1.x utilities are anything to go by, the V2 ones should be very interesting indeed.

Have you upgraded yet?

For Your Diary

Here's a list of the meetings taking place over the next few months. You'll be thrilled to see a quiz (don't tell C**m) appearing before the summer meetings—should be good for a laugh (or two) and a prize.

1 May	An ocean of (Fred) fish—now on CD-ROM
15 May	It's a table quiz—no brains required
29 May	An introduction to ARexx
12 June	To be announced
26 June	Treasurer's holiday photos
10 July	Last chance to submit articles for July Newsletter
24 July	Last meeting before the summer break

Please note that the items advertised may be subject to change without notice. Associate members should not expect the usual prompt (!) replies to correspondence for the next few months, as some committee members may be on holiday.

Camera ready pages for this issue
were produced using:

Amiga 3000/25-100
AmigaTEX V3.1h
Sharp Laser Printer

Photocopied by
whoever was available